Introducing Cloud Computing Topics in Curricula

Ling Chen
Yang Liu
Marcus Gallagher
School of Information Technology and Electrical Engineering
University of Queensland
Brisbane, QLD 4072, Australia
l.chen5@uq.edu.au, uqyliu19@uq.edu.au,
marcusg@itee.uq.edu.au

Bernard Pailthorpe
School of Mathematics and Physics
University of Queensland
Brisbane, QLD 4072, Australia
b.pailthorpe@uq.edu.au

Shazia Sadiq
Heng Tao Shen
Xue Li
School of Information Technology and Electrical Engineering
University of Queensland
Brisbane, QLD 4072, Australia
shazia@itee.uq.edu.au, shenht@itee.uq.edu.au,
xueli@itee.uq.edu.au

ABSTRACT

The demand for graduates with exposure in Cloud Computing is on the rise. For many educational institutions, the challenge is to decide on how to incorporate appropriate cloud-based technologies into their curricula. In this paper, we describe our design and experiences of integrating Cloud Computing components into seven third/fourth-year undergraduate-level information system, computer science, and general science courses that are related to large-scale data processing and analysis at the University of Queensland, Australia. For each course, we aimed at finding the best-available and cost-effective cloud technologies that fit well in the existing curriculum. The cloud related technologies discussed in this paper include open-source distributed computing tools such as Hadoop, Mahout, and Hive, as well as cloud services such as Windows Azure and Amazon Elastic Computing Cloud (EC2). We anticipate that our experiences will prove useful and of interest to fellow academics wanting to introduce Cloud Computing modules to existing courses.

Keywords: Cloud Computing, Data-intensive, MapReduce, Hadoop

1. INTRODUCTION

Cloud computing is a computing model that provides modern on-demand services for management and usage of large shared resources including storage, computations and communications. Cloud technologies, especially the MapReduce framework (Dean and Ghemawat, 2004) and its open-source implementation Hadoop (Hadoop website, 2012), enable scalable distributed processing of huge amounts of data. To develop and manage information systems for large scale data analysis and processing, cloud computing related knowledge and skills are becoming fundamental for computing professionals.

Many leading universities have recently offered cloud-related formal subjects (TCSC website, 2010). Several schools have incorporated Hadoop into their curricula. For example, Rabkin et al. (2012) at UC Berkeley have integrated four two-week comprehensive Hadoop assignments into an introductory machine organization course. St. Olaf College (Brown, 2009) has built clusters to run Hadoop based on retired computers for students in computer science courses. Later, they constructed virtual clusters based on lab machines for parallel and distributed computing courses (Shoop et al., 2012). Wang et al. (2011) introduced cloud computing with a senior design project for IT undergraduate students. Many works used cloud technologies to improve students' learning experiences. For example, Vaquero (2011) did a high quality case study in comparing different cloud service mode usages in an advanced computer science course.

This paper describes how we have integrated cloud technologies into seven existing courses at the University of Queensland (UQ), based on a collective effort of colleagues in a Teaching and Learning Project. This work is different from the previous works in the following aspects. Firstly, it covers a variety of courses ranging from data management and analysis, high performance computing to scientific computing and visualization, while previous works mostly focus on a single course, or a particular cloud related technology (e.g., Hadoop). Thus, the challenge we faced lies in how to choose the best available cloud technologies that fit well in the existing course curriculum and its assumed background. Secondly, a cloud module in our case serves as a small module for existing courses, typically for a week or two. Therefore, the goal is to provide students with hands-on cloud experiences as a useful problem-solving tool in the field covered by the course, rather than an in-depth introduction to the computing model itself or one of its related technologies. Finally, our work has covered some additional Hadoop-based tools such as Hive (Hive website, 2012) and Mahout (Mahout website, 2012), as well as Windows Azure cloud services (Windows Azure website, 2012).

The significance of this paper is twofold. Firstly, it shows how we have achieved this integration with minimum budget overhead. Secondly, it shares the development experiences and considerations that can be helpful for fellow academics who are interested in integrating cloud computing into existing courses.

The rest of the paper is organized as follows. Section 2 gives an overview of the cloud computing model and our preliminary experiments on Hadoop, an open-source implementation of the MapReduce framework. Section 3 reports the detailed cloud practical exercise designs for seven targeted courses, followed by the conclusion in Section 4.

2. CLOUD COMPUTING MODEL: AN EMERGING IT INFRASTRUCTURE

2.1 Cloud Computing

Cloud computing is a computing model built on decades of research in virtualization, distributed computing, utility computing, and networking. As a computing model, it provides the modern on-demand services for management and usage of large shared computing resources including storage, computations, and communications. The level of abstraction it offers allows end-users to access to IT-enabled services via the Internet "as a service," without the knowledge of or control over the underlying infrastructure that supports them. Currently there is no consensus on its definition in the academic world, but researchers often adopt the version of the U.S. National Institute of Standards and Technology (Mell and Grance, 2011).

The most popular cloud service modes are *Software as a Service* (SaaS), *Platform as a Service* (PaaS), and *Infrastructure as a Service* (IaaS) (Mell and Grance, 2011). Software as a Service gives access to applications such as email client (e.g., GMail), text editor (e.g., Google Docs), or online storage (e.g., Dropbox). Platform as a Service provides a container environment to run user application components. For instance, users can deploy their own .NET Web applications onto Windows Azure cloud instances (Windows Azure website, 2012). Infrastructure as a Service supplies infrastructural resources, usually in terms of virtual machines, upon which consumers can deploy and run arbitrary software including operating systems and applications. Amazon Elastic Compute Cloud (Amazon EC2 website, 2012) is one of the well-known IaaS vendors.

In order to process huge data efficiently, the MapReduce framework, originally proposed by Google Research (Dean and Ghemawat, 2004) with its underlying distributed file system (Ghemawat et al., 2003), is one of the most popular cloud technologies applied to cloud services or computer clusters. The idea comes from functional programming where users can define a map and a reduce function (Lämmel, 2008). In MapReduce, a Map function takes data sets in key-value pair format and generates a set of intermediate key-value pairs. A Reduce function then merges all intermediate pairs with the same key. The model follows the master-slave architecture where computing jobs are submitted to the Master machine by the user program. The Master machine firstly partitions input data into *M* Map tasks or "splits" and allocates them to Map workers to process with Map function. Then Reduce workers merge intermediate results output by Map workers using the Reduce function. Sometimes there is an extra Combiner for combining all the intermediate results from Reduce workers. MapReduce utilizes loosely-coupled parallelism to handle data-intensive jobs. Loosely-coupled parallelism works on computing tasks that are breakable into independent parallel parts with very minimal or no data synchronization among each other. In contrast, tightly-coupled parallelism often requires communications between the different parallel tasks. One important assumption of MapReduce is the use of commodity machines that can easily fail. As a result, it has implemented fault-tolerance mechanism to handle machine failures.

We chose Hadoop because it is one of MapReduce's most widely used open-source implementations, though Microsoft Research has also developed their DryadLINQ (Yu et al., 2008). The core components of Hadoop are Hadoop MapReduce and Hadoop Distributed File System (HDFS). Hadoop ecosystem contains a range of Hadoop extensions for particular problem domain. For example, Hive is a Hadoop-based data warehousing tool (Hive website,

2012) and Mahout is a machine learning tool that runs classical learning algorithms in parallel using Hadoop (Mahout website, 2012).

2.2 Preliminary Experiments

Currently Hadoop is running on a Unix Server called "moss" that is accessible for all students enrolled in relevant courses. This Hadoop cluster has 400 nodes built on lab machines of the School of Information Technology and Electrical Engineering (ITEE) at UQ. At the stage of investigating Hadoop, we built a tiny cluster for experiments in order to measure efficiency improvement by increasing the number of nodes in the cluster.

We used the MEDLINE data set in our experiments. It is the premier bibliographic database of the U.S. National Library of Medicine (2012). The data set is in XML (Extensible Markup Language) format and the compressed files are freely downloadable from its website. We used a subset of size 50.7 GB. The task is to find articles associated with the topic "biology." This task requires one scan over the entire data set and retrieving all the records that satisfy the search condition. We ran Hadoop on this task with different number of machines (i.e., 1, 2, 3, and 4), in order to observe the time performance improvement with the increase of number of machines. The scale of the cluster is tiny, i.e., only 4 PCs with Ubuntu 10.10 (Linux) and hadoop-0.20.203.0 installed. Table 1 shows the specifications of the PCs that form the cluster. They are the so-called "commodity machines," old and with humble computing power and memory.

Machine Name	Model	CPU	RAM	Disk
Master	DELL OPTIPLEX 960	2.0GHz	3GB	120GB
Slave 1	DELL OPTIPLEX 280	2.0GHz	1.5GB	80GB
Slave 2	DELL OPTIPLEX 520	2.0GHz	2GB	80GB
Slave 3	DELL OPTIPLEX 520	2.0GHz	1GB	80GB

Table 1: Machine Specification

Figure 1 shows that adding more machines to share the workload in the MapReduce framework has significantly reduced the computing time. With only one machine, it took about 71 minutes to complete the task; however, with four machines, the execution time reduced significantly to about 40 minutes, nearly half of the time spent with one machine. From the teaching and learning point of view, the significance of this experiment is that even with limited number of commodity machines, students can still experience the power of a MapReduce-based distributed computing framework like Hadoop.

3. DESIGNS OF CORE COURSES

We have identified seven UQ courses related to large-scale data processing, analysis and visualization as the core courses. They are mainly information system courses and scientific computing related courses. The outcome cloud materials include lecture notes, tutorial questions and answers, and practical exercises. This paper focuses on the practical exercises, as the goal is to develop technology-oriented cloud modules.

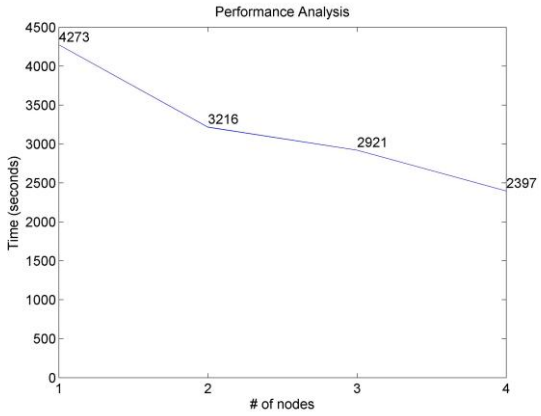


Figure 1: Performance of Different Number of Nodes

We have included a mapping table in Figure 2 to illustrate the mapping between cloud technologies and seven core courses. The table on the right lists the courses and the corresponding course components that can be possibly the starting points for introducing cloud technologies. The table in the middle suggests the possible cloud components for the course components. Finally, the table on the left displays the required cloud services and technologies for implementing those cloud components. As a general rule, lecture notes cover basic cloud definitions, associated cloud services and technologies, pros and cons, issues, and applications, and introduce them from the course's perspective. For example, cloud modules for scientific computing related courses approach cloud concepts from the aspect of scientific applications, while in the course of Service-oriented Architecture (SOA), a cloud service serves as an external service in SOA.

The main challenge lies in how to design a cloud module aligned with the objectives of each course. This includes finding the best available and cost effective cloud technologies, as well as developing practical exercises suitable for a maximum of two week's schedule with a reasonable level of difficulty. We have classified core courses into three categories: data processing and database management, high performance computing, and scientific computing and visualization.

3.1 Data Processing and Database Management

Table 2 lists three information system courses belonging to the data processing and database management category.

Course Code	Course Name	Size
INFS3200	Advanced Database Systems	70-100
INFS3202	Web Information Systems	100-130
INFS3204	Service Oriented Architectures	70-90

Table 2: Data Processing and Database Management Courses

3.1.1 EC2 Web page for INFS3202: Web Information Systems is an information system course that covers a range of scripting languages for Web design such as HTML, CSC,

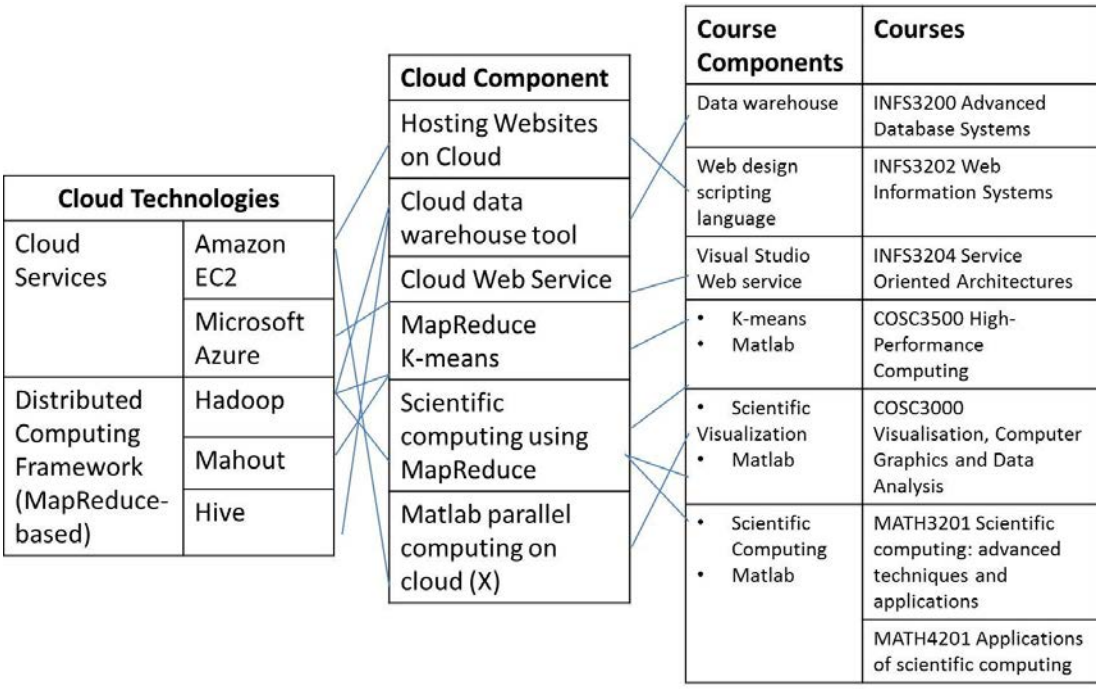


Figure 2: Mapping of Courses and Cloud Technologies

PHP, JavaScript, JSP, etc. In weekly practical sessions, students gradually build up a Web information system by adding a new technology learned in each week. For this reason, we designed the practical exercises for the cloud module as hosting on EC2 cloud service a website created in the previous practical sessions. This EC2 practical exercise contains four steps.

- Register for an Amazon Web Service (AWS) account.
- Create an EC2 Linux micro instance (of the Free Usage Tier).
- Upload the Web application files via a FTP (File Transport Protocol) onto the EC2 instance.
- Deployed a website onto the cloud instance.

Figure 3 displays a screenshot of our cloud project website hosted on EC2, as an example in the practical guide.

3.1.2 Azure cloud service for INFS3204: The course Service Oriented Architectures (SOA) aims to give students comprehensive understanding of basic service-oriented concepts and architecture, as well as practical experiences of the architecture using Microsoft Visual Studio 2010 .NET environment. The weekly practical exercises help students gradually build up a Web information system by creating and using both internal and external services. To introduce the cloud module into SOA, we incorporate cloud components as an external service hosted on cloud provider's machines. We chose Microsoft's cloud platform Windows Azure based on the compatibility concerns (i.e., both Visual Studio 2010 and Windows Azure are Microsoft products) as well as the advantage that Windows Azure academic free pass is currently available for universities to apply for teaching purpose. So, this module is a Web service deployed on a Windows Azure cloud instance, callable by client applications. It requires Windows 7 operating system (OS) and Visual Studio 2010 (VS2010) and contains the following steps. A graphic overview can be found in Figure 4.

- Download the Windows Azure SDK (Software Development Kit) for VS2010 for creating a cloud Web service directly from the VS2010 cloud template.
- Create an Azure account and a new hosted service on Azure. Choose a Web service created in the previous practical session and recreate it as a cloud service

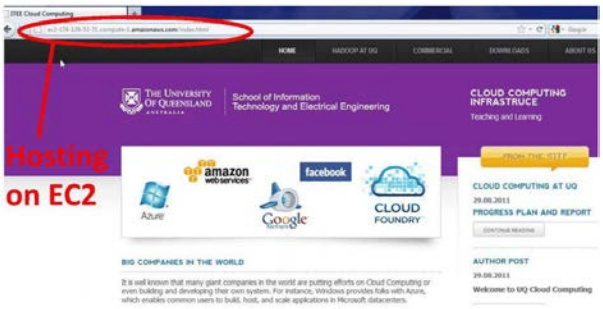


Figure 3: Web Page Hosted on EC2

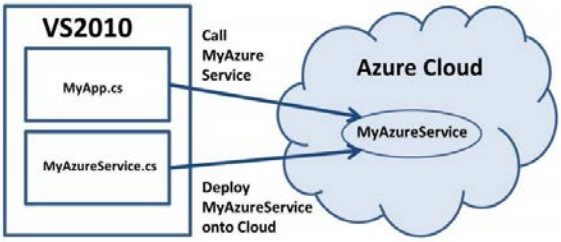


Figure 4: Azure Practical Overview

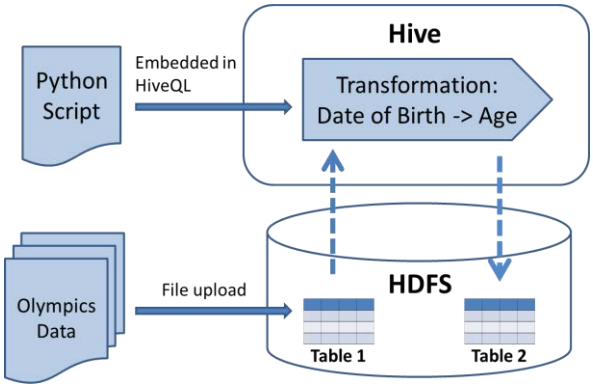


Figure 5: Overview of Hive Practical Design

using the VS2012 cloud template.

- Deploy the cloud service onto the cloud instance.
- Call the cloud service from application code as an external service.

3.1.3 Hive for INFS3200: Advanced Database Systems is a third-year database system course at UQ. As an advanced course, it assumes that students already have the theoretical knowledge on the Relational Model and practical experiences in Relational Database Management Systems (RDBMS) like Oracle, as well as its query language SQL. The course introduces a range of advanced database-related topics, such as distributed database architectures, data warehouse, data intensive applications, sensor networks, etc. So, the question for us is how to introduce Cloud technologies in a way that demonstrates its usefulness in existing topics?

A typical problem for analyzing huge data storing in RDBMS as well as data warehouse is the efficiency issue for query processing. This motivated us to introduce a cloud-based data warehousing tool. Hive (Thusoo et al., 2009) is a Hadoop-based data warehousing tool designed to handle large data analysis. It uses HiveQL, a query language very similar to the standard SQL that students are familiar with. HiveQL allows traditional data ETL (Extract, Transform and Load) to be embedded in HiveQL statements.

Since the course does not assume any Linux background, our original design followed what Windows users would do if they wish to run Hadoop-based programs on Windows OS, i.e., setting up Cygwin (Cygwin website, 2012), a Linux-like environment sitting upon Windows OS. However, we soon realized that even with Cygwin, students still have to learn Linux terminal commands in order to run Hadoop and Hive. So, we decided to give students a real UNIX/Linux environment experience, i.e., running Hive on school's "moss" server. The main task of this Hive practical exercise is to use the embedded Python script to transform the dates of birth of Olympics athletes into their ages and store the results into another table. Figure 5 gives an overview of our design.

The practical exercise comprises four steps:

- Create tables and load data. Students are instructed to create tables with partitions using PARTITIONED BY clause in the HiveQL CREATE TABLE statement.
- Upload an Olympics dataset for athletes (DatabaseOlympics website, 2011) onto Hadoop Distributed File System (HDFS).
- Use HiveQL data ETL function with Python script. We have created a Python script for students to compute the ages of Olympics athletes from their dates of birth. An example of such a Python script is given in Figure 6. Then, they can issue a HiveQL command with the TRANSFORM operator and USING clause to run the script. This command takes input data from one table and stores the transformed results into another table.
- View the results via HDFS Web interface at http://namenode:50070.

3.2 High-Performance Computing

COSC3500 High-Performance Computing (HPC) is part of the computational science program in the Bachelor of Science at UQ and is also accessible to interested students in other programs. The course primarily focuses on the practical applications of HPC technologies to solving problems across all fields of science. An important component of the course is parallel computing, using Message Passing Interface (MPI) and Open Multiprocessing

```
import time,datetime
import sys

curDate= datetime.datetime(*(time.localtime(time.time()))[0:6])

for line in sys.stdin:
    red=""
    line= line.strip()
    AthleteID, FirstName, LastName, DOB, Gender, Country= line.split('\t')
    if DOB!="":
        date1= time.strptime(DOB, "%Y/%m/%d %H:%M:%S")
        date2= datetime.datetime(*date1[0:6])
        red= curDate- date2
    print ",".join([AthleteID, FirstName, LastName, DOB, str((red.days)/365), Gender, Country])

    else:
        red="NULL"
    print ",".join([AthleteID, FirstName, LastName, str(red)])
```

Figure 6: Python Script for Hive

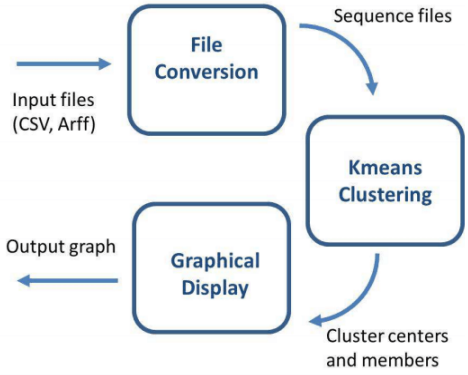


Figure 7: Mahout Practical Overview

(OpenMP). They belong to the category of tightly-coupled parallelism.

To facilitate the comparison between tightly-coupled and loosely-coupled parallelisms, we have designed a Hadoop-based module for this course, as Hadoop MapReduce is a realization of loosely-coupled parallelism. The practical exercise uses Mahout, a Hadoop-based machine learning tool that runs classical machine learning algorithms in parallel with Hadoop MapReduce. Particularly, we have chosen the classical K-means clustering algorithm because it is a relatively compact and simple algorithm, and it can also be implemented with MPI and OpenMP (Liao, 2009) for the purpose of comparison.

K-means algorithm (MacQueen, 1967) is a clustering algorithm that iteratively reassigns each data point to the closest cluster and updates the cluster means (i.e., centroids) until the cluster means converge. The input data are multi-dimensional points expressed as vectors. The KMeansDriver Java class in Mahout API (Application Programming Interface) firstly splits the input data into subsets and sends them to Map workers called KMeansMapper. The output of a Mapper is a set of Key-Value pairs ‹Cj,Vi› such that Cj is the cluster ID whose centroid is closest to point Vi than any other cluster centroids. Then, KMeansCombiner, an extra Combiner merges all the pairs with the same key, i.e., Cj. Finally, KMeansReducer computes new cluster centroids based on outputs from KMeansCombiner.

As currently "moss" server does not yet support Mahout, the practical exercise guides students to install Hadoop 0.20.203 and Mahout 0.5 on an Ubuntu 10.10 OS, as other components of the course also assume a Linux environment. Figure 7 shows an overview of the design. For the input data set, we use the first two attributes of the well-known Iris data set from UCI Machine Learning Repository (Frank and Asuncion, 2010). The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. Only one class among the three is linearly separable from the other two. Students can obtain Iris data set in ARFF format directly from the distribution of WEKA, a well-known open-source machine learning software (Hall et al., 2009).

Firstly, students have to convert the input files into sequence files, because Mahout algorithms take only Hadoop sequence file format as input. Mahout's terminal command

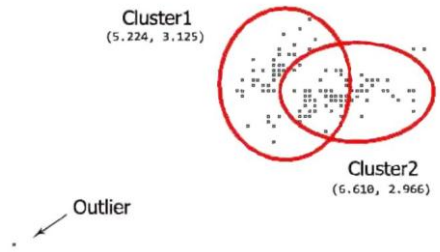


Figure 8: Mahout Clustering on Iris Data Set

"bin/mahout arff.vector" converts a file from ARFF (Attribute-Relational File Format) to a sequence file. Mahout also supports the conversion from CSV (Comma-separated values) file, but only via Mahout Java API. After the file conversion, students are ready to run K-means with "kmeans" command using different parameter settings, such as distance measurements, threshold of convergence, and number of clusters.

In the practical guide, these parameters are set as Euclidean distance, 2 and 0.001. Finally, students can view the results by running a jar file we created for displaying clustering results based on the DisplayClustering class from Mahout API. Figure 8 is a screenshot of running the jar file, where clustering results are dots in circles with cluster centers specified under the cluster names.

3.3 Scientific Computing and Visualization

Table 3 lists the courses in this category. They are not core computer science courses, but they aim to assist science students to utilize computational technologies for scientific analysis. As these courses assume Linux OS, we use Ubuntu 10.10 for Hadoop-based practical exercises.

Course Code	Course Name	Size
MATH3201	Scientific Computing: advanced techniques and applications	15-20
MATH4201	Applications of Scientific Computing	5-10
COSC3000	Visualization, Computer Graphics and Data Analysis	10-20

Table 3: Scientific Computing and Visualization Courses

3.3.1 Scientific Computing: Scientific computing is interdisciplinary in nature. With the growing computing power and storage, computer simulation and modeling are becoming powerful tools for a variety of science and engineering disciplines. Scientific computing communities are well-aware of the need to process large amounts of collected or simulated data more efficiently. For the last two decades, many researchers have devoted themselves to Grid computing and shown great works (Foster and Kesselman, 2003). Some university courses also adapted Grid for teaching scientific computing. For example, (Löffler et al., 2011) has incorporated TeraGrid in their scientific computing course at Louisiana State University. The increasing popularity of cloud computing has drawn these

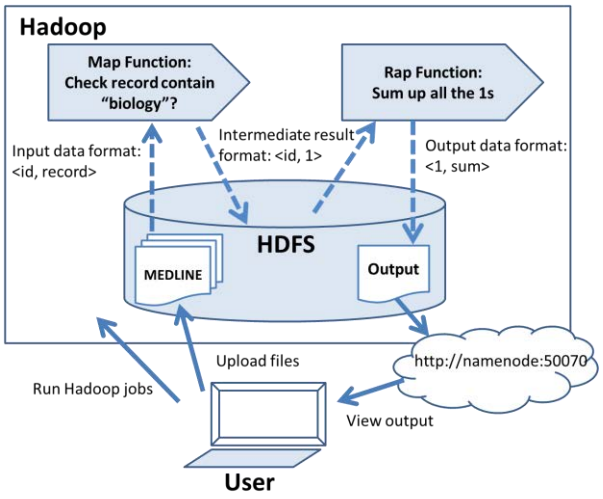


Figure 9: Hadoop Practical on MEDLINE Data Set

communities to consider Cloud (Foster et al., 2008). The question for teachers in scientific computing courses is then, how to incorporate cloud technologies into the curriculum?

MATH3201 at UQ is an introduction course to scientific computing. It covers aspects of scientific computing ranging from numerical analysis, high performance computing, and grid computing to scientific visualization. The subsequent course MATH4201 has a special focus on numerical modeling, for analysis of complex biological, physical and engineering systems. It covers scientific simulation packages such as MATLAB, Mathematica, Python, parallel programming, and aspects of HPC.

Our first choice for practical exercises was MATLAB parallel computing on Amazon Elastic Compute Cloud (EC2) cloud service, as both MATH3201 and MATH4201 courses use MATLAB as one of its numerical modeling and visualization tools. However, running MATLAB parallel computing on Amazon EC2 requires the Distributed Computing Server from MathWorks for distributing and monitoring jobs, but unfortunately it is unaffordable. As a result, we started to consider the possibility of leveraging the loosely-coupled parallelism offered by the Hadoop MapReduce framework.

Given the limited time allocated to the cloud module of MATH3201, the goal is for students to learn the loosely-coupled parallelism provided by Hadoop. Thus the practical exercises mainly focus on how to set up Hadoop and run a simple job on it. The exercise contains three parts:

- Hadoop installation and configuration on a single-node Hadoop cluster.
- Word count example: this includes uploading text files onto HDFS and using the command line to run hadoop-examples-*.jar that comes with the Hadoop distribution. The "wordcount" method counts the occurrences of each word appears in the text files.
- View Hadoop statistics, such as file size, number of successful/failed Map and Reduce tasks, etc. This information is accessible at http://namenode:50030 by default, where "namenode" is the IP address of the Master machine.

```
<MedlineCitation Owner="NLM" Status="MEDLINE">
    <PMID Version="1">22814699</PMID>
          ⋮
    <Article PubModel="Print">
        <Journal>
            <ISSN IssnType="Print">1432-8798</ISSN>
            <JournalIssue CitedMedium="Print">
                ⋮
            </JournalIssue>
            <Title>Archives of Virology</Title>
        </Journal>
        <ArticleTitle>Biology of fowl adenovirus ty
                ⋮
</MedlineCitation>
```

Figure 10: MEDLINE Record in XML

- Build a multi-node cluster and run the same task again to see the performance improvement. To do this, we divided students into groups of 3-5 students per group and instructed them to build a multi-node cluster by changing some settings in their single node cluster.

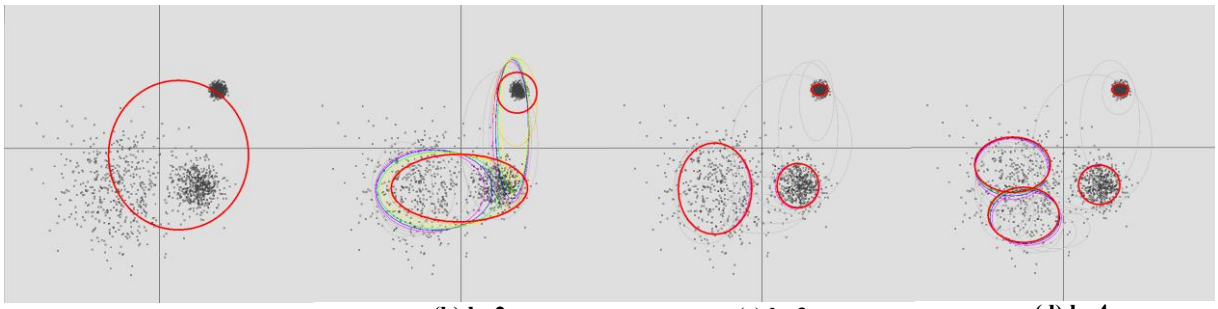
MATH4201 is an advanced Scientific Computing course, so we can reasonably assume that students have taken MATH3201 and learned how to set up a Hadoop cluster. So, this time they need to write their own Mapper and Reducer in Hadoop and run it on school's 400 node "moss" server. As the assumed programming language in the course is Python and not Java, we chose to use Hadoop streaming, a Hadoop utility that allows users to run Map/Reduce jobs with any executable or scripts as Mapper and/or Reducer.

The Hadoop practical exercise for MATH4201 is an application of scientific computing in the medical science domain. It is similar to the task in our Hadoop experiments described earlier. The idea is to guide students to think how to define Mappers and Reducers to be called by multiple data nodes in parallel. Figure 9 shows an overview of the design.

The exercise contains four steps:

- Choice of data set: the MEDLINE data set is freely downloadable from the website of the U.S. National Library of Medicine (2012), but students may choose other data sets.
- Python Mapper and Reducer script: we provide Python Mapper and Reducer scripts as templates. Our Mapper script reads a MEDLINE record at a time. It gets the value from the ‹ArticleTitle› tag of the record as shown in Figure 10 and check if it contains the query term (e.g., "biology") as a substring. The Reducer script then sums up the number of key-value pairs from the Mappers, by counting the lines of input data, and outputs the final sum.
- Run Hadoop MapReduce and view the results: finally, students can run the scripts with input data using Hadoop streaming and view the results on HDSF Web interface at http://namenode:50070, where "namenode" is the IP address of the Master machine.

3.3.2 Scientific visualization: Scientific visualization allows scientists to create graphical representations of the data obtained by scientific instruments or by the output from computations and simulations. However, the collected or



(a) k=1 (b) k=2 (c) k=3 (d) k=4

Figure 11: Clustering Results with k=1, 2, 3, and 4

created data are often too huge to process in reasonable amount of time. Researchers in the Grid computing community have many sophisticated works on grid-enabled visualization (Ahmed et al., 2007). By contrast, there are so far only a few works on cloud-based visualization, mostly using the MapReduce framework (Nguyen et al., 2011; Tanahashi et al., 2010; Vo et al., 2011).

COSC3000 is an introductory course in computer graphics and scientific visualization. Students learn how to do 2D and 3D plotting, data transformation, rotation, animation, etc. using tools such as MATLAB, Mathematica, ParaView (a GUI to the Visualization Toolkit), Python and OpenGL, and apply this knowledge to project as part of the assessment. The challenge for developing a cloud module for this course is that cloud-based visualization is a rather new research area and the frameworks descripted in existing works are often too complicated to adapt (Nguyen et al., 2011; Tanahashi et al., 2010; Vo et al., 2011). As a result, our design for the practical exercise focuses on cloud-based clustering, as clustering is a classical approach for grouping data and visualizing data (especially for 2 or 3 dimensions).

In the cloud practical design, the goal is for students to learn how to use Hadoop-based clustering tools and visualization for data analysis. A Java template provided to students modifies DisplayClustering class in Mahout. We developed a template based on Mahout API for students to generate points from a number of distributions given sample mean and standard deviation. Then the instructions ask students to compare and discuss on different parameter settings with the corresponding results. For example, Figure 11 (a)-(d) show the K-means clustering results for k = 1, 2, 3, 4 respectively. The 2D data points are drawn from three Normal distributions with sample mean and standard deviation settings (‹-1,-1›, 0.8), (‹1,-1›, 0.3) and (‹1.5,1.5›, 0.1). As the "optimal" number of clusters k is unknown in most clustering situations, the example allows students to observe how the choice of k affects the clustering results given the known distribution. Each color in Figure 11 indicates the results after a clustering iteration and the thick red circles mark the final clusters. This example also demonstrates how visualization tools can help for understanding and analyzing the data.

4. CONCLUSIONS

The increasing popularity of cloud computing and the growing awareness of its potential to handle "big data" and to reduce IT-related costs have gradually paved its way to university curricula. This paper demonstrated how we have introduced cloud computing into seven existing information system, computer science and general science courses at the University of Queensland. These core courses are third or fourth-year courses related to large-scale data processing, analysis and visualization. A typical cloud module includes lecture notes, tutorial questions and answers, and practical exercises. We have illustrated the rationale and design considerations for these cloud modules, with an emphasis on practical exercise designs.

The cloud related technologies covered in the outcome materials include the Amazon EC2 and Window Azure cloud services, the Hadoop core package for distributed computing, and other Hadoop-based tools such as Mahout and Hive. These technologies are employed for hosting websites and external services, parallel data processing and handling of large data sets, and scientific computing and visualization. Our Hadoop experiments on a 50.7 GB data set has shown that students can easily observe significant reduction of computational time by running Hadoop with only four commodity machines.

The significance of this work is twofold. Firstly, it shares our rationale and considerations of choosing cloud-related technologies for courses ranging from data analysis and management, high performance computing, to scientific computing and visualization. This differs from designing a course specialized in cloud computing or a specific topic. The main challenge lies in how to find the best available cloud computing related technologies that align with course objectives and fit well into other components of the course. Secondly, it shows how this can be achieved with limited budget and resources, as key technologies employed in our practical exercise designs are mostly open-source and/or free.

5. ACKNOWLEDGEMENTS

This work was supported by the Teaching and Learning Grant from Faculty of Engineering, Architecture and Information Technology, the University of Queensland, Australia.

6. REFERENCES

Ahmed, A. A., Latiff, M. S. A., Abu Bakar, K. and Rajion, Z. A. (2007) "Visualization Pipeline for Medical Datasets on Grid Computing Environment," International Conference on Computational Science and its

Applications (ICCSA'07), Ho Chi Minh City, Vietnam, pp. 567-576.

Amazon Elastic Compute Cloud (EC2) (2012), Retrieved July 30, 2012, from http://aws.amazon.com/ec2/.

Brown, R. A. (2009) "Hadoop at Home: Large-scale Computing at a Small College," SIGCSE Bull., Vol. 41, No. 1, pp. 106-110.

Cygwin (2012), Retrieved July 30, 2012, from http://www.cygwin.com/.

Dean, J. and Ghemawat, S. (2004) "MapReduce: Simplified Data Processing on Large Clusters," Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation, San Francisco, CA, pp. 137-149.

Foster, I. and Kesselman, C. (2003) The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers Inc., San Francisco, CA.

Foster, I., Zhao, Y., Raicu, I. and Lu, S. (2008) "Cloud computing and grid computing 360-degree compared," IEEE Grid Computing Environments Workshop, Austin, Texas, pp. 1-10.

UCI Machine Learning Repository (2010), Retrieved July 30, 2012, from http://archive.ics.uci.edu/ml.

Ghemawat, S., Gobioff, H. and Leung, S. T. (2003) "The Google File System," Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03), Bolton Landing, New York, pp. 29-43.

Hadoop (2012), Retrieved July 30, 2012, from http://hadoop.apache.org/.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H. (2009) "The WEKA Data Mining Software: An Update," SIGKDD Exploration, Vol. 11, No. 1, pp. 10-18.

Hive (2012), Retrieved July 30, 2012, from http://archive.ics.uci.edu/ml.

Lämmel, R. (2008) "Google's MapReduce Programming Model - Revisited," Science of Computer Programming, Vol. 70, No. 1, pp. 1-30.

Parallel K-Means Data Clustering (2009), Retrieved July 30, 2012, from http://users.eecs.northwestern.edu/~wkliao/Kmeans/index.html.

Löffler, F., Allen, G., Benger, W., Hutanu, A., Jha, S. and Schnetter, E. (2011) "Using the TeraGrid to teach scientific computing," Proceedings of the 2011 TeraGrid Conference: Extreme Digital Discovery, Salt Lake City, Utah, pp. 1-7.

MacQueen, J. (1967) "Some Methods for Classification and Analysis of Multivariate Observations," Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, pp. 281-297.

Mahout (2012), Retrieved July 30, 2012, from http://mahout.apache.org/.

Nguyen, A. V., Wynden, R. and Sun, Y. (2011) "HBase, MapReduce, and Integrated Data Visualization for Processing Clinical Signal Data," 2011 AAAI Spring Symposium Series, Palo Alto, California.

Rabkin, A. S., Reiss, C., Katz, R. and Patterson, D. (2012) "Experiences Teaching MapReduce in the Cloud," Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, Raleigh, North Carolina, pp. 601-606.

Shoop, E., Brown, R., Biggers, E., Kane, M., Lin, D. and Warner, M. (2012) "Virtual Clusters for Parallel and Distributed Education," Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, Raleigh, North Carolina, pp. 517-522.

Tanahashi, Y., Chen, C. K., Marchesin, S. and Ma, K. L. (2010) "An Interface Design for Future Coud-based Visualization Services," IEEE 2nd International Conference on Cloud Computing Technology and Science (CloudCom), Indianapolis, Indiana, pp. 609-613.

IEEE TCSC Cloud Computing Courses (2010), Retrieved July 30, 2012, from http://sites.google.com/site/cloudcomputingsystem/course.

Thusoo, A., Sarma, J. S., Jain, N., Shao, Z., Chakka, P., Anthony, S., Liu, H., Wyckoff, P. and Murthy, R. (2009) "Hive: A Warehousing Solution over a Map-Reduce Framework," Proceedings of the VLDB Endowment, Vol. 2, No. 2, pp. 1626-1629.

MEDLINE Fact Sheet (2012), Retrieved July 30, 2012, from http://www.nlm.nih.gov/.

Vaquero, L. M. (2011) "EduCloud: PaaS versus IaaS Cloud Usage for an Advanced Computer Science Course," IEEE Transactions on Education, Vol. 54, No. 4, pp. 590-598.

Vo, H. T., Bronson, J., Summa, B., Comba, J. L. D., Freire, J., Howe, B., Pascucci, V. and Silva, C. T. (2011) "Parallel Visualization on Large Clusters Using MapReduce," IEEE Symposium on Large Data Analysis and Visualization (LDAV) Seattle, Washington, pp. 81-88.

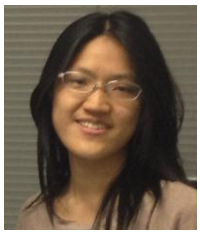
Wang, X., Hembroff, G. C., Cerier, A. B. and Perrault, B. W. (2011) "Introducing Cloud Computing with a Senior Design Project in Undergraduate Education of Computer System and Network Administration," Proceedings of the 2011 Conference on Information Technology Education, West Point, New York, pp. 177-182.

Windows Azure (2012), Retrieved July 30, 2012, from http://www.windowsazure.com/en-us/.

Yu, Y., Isard, M., Fetterly, D., Budiu, M., Erlingsson, Ú., Gunda, P. K. and Currey, J. (2008) "DryadLINQ: A System for General-purpose Distributed Data-parallel Computing Using a High-level Language," Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation, San Diego, California, pp. 1-14.

AUTHOR BIOGRAPHIES

Ling Chen is a Research Assistant in the School of Information Technology & Electrical Engineering (ITEE) at the University of Queensland (UQ). She obtained her Master's degree in Computer Science from the University of Queensland in 2011. Her research interests include Information Network Analysis, Machine Learning, and Data Mining.



Yang Liu is a Research Assistant in the School of Agriculture and Food Science at the University of Queensland (UQ). He obtained his Master's degree in Information Technology from The University of Queensland in 2011. His research interests include Big Data Retrieval, Cloud & Parallel Computing and Artificial Intelligence.

Marcus Gallagher is Senior Lecturer in the School of Information Technology & Electrical Engineering (ITEE) at the University of Queensland (UQ). He obtained his PhD from the University of Queensland in 2000. He also completed a GradCert (Higher Education) in 2010. His research interests include Optimization, Metaheuristics and Evolutionary Computation, Machine Learning and Exploratory Data Analysis and Visualization

Bernard Pailthorpe is a Professor of Computational Science in the School of Physical Sciences and a Director of the Research Computing Center at the University of Queensland (UQ). His research interests are Statistical Mechanics, Molecular Dynamics, Scientific Computing, Scientific Data Visualisation, Access Grids and Tiled Displays.

Shazia Sadiq is a Professor in the School of Information Technology & Electrical Engineering (ITEE) at the University of Queensland (UQ). She obtained her PhD in Information Systems from The University of Queensland. Her research interests include Business Process Management, Governance, Risk and Compliance, Data Quality Management, Workflow Systems, and Service Oriented Computing.

Heng Tao Shen is a Professor of Computer Science and an ARC Future Fellow in the School of Information Technology & Electrical Engineering (ITEE) at the University of Queensland (UQ). He obtained his PhD from Department of Computer Science, National University of Singapore (NUS) in 2004. His research interests are Multimedia, Mobile and Web Search, and Big Data Management.

Xue Li is an Associate Professor in the School of Information Technology & Electrical Engineering (ITEE) at University of Queensland (UQ). He obtained his PhD degree in Information Systems from Queensland University of Technology (QUT) in 1997. His research interests include Data Mining, Multimedia Data Security, Database Systems, and Intelligent Web Information Systems.

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096